Abstract

This document contains the slides for the keynote speech at the second day of the 2nd International Workshop on Feature Interaction in Telecommunications Systems, 9-10 May, 1994, Amsterdam, The Netherlands.
Using an Architecture to Help Beat Feature Interaction

Rob van der Linden

ANSA Project Manager

rvdl@ansa.co.uk
Background
Main Themes

- separation and substitutability
- available and accessible descriptions
Meet the customer’s needs

- **timely:** I want it NOW! (well, yesterday)
- **personalised:** I want it just the way I like it!
- **competitive:** I don’t want to pay over the odds!
- **dependably predictable:** It should be there when I need it!
- **integratable:** Why can’t I connect it to my PC?

. . . . . . before your competitor does
The business challenge

Customer’s Willingness to Pay

Revenue that must be acquired through value added services

Cost of providing Basic Transport

$
Software cost

capacity/unit cost
(log scale)

Hardware
Software

A new problem?

• project management
• requirement analysis
• analysis and design methodologies
• informal and formal approaches

No, but none of these SCALE very well
System structure: separation

Minimum shared communications and processing resources
System structure: Substitutability (1)

- operational Interfaces

```
Interface {
    operation₁Name ( < typed parameter > )
    -> outcome₁Name ( < typed result > )
    :
    -> outcomeₙName ( < typed result > );
    :
    operationₙName ( < typed parameter > )
    -> outcome₁Name ( < typed result > )
    :
    -> outcomeₙName ( < typed result > );
};
```
System structure: Substitutability (2)

- **stream Interfaces**
  - for computationally unstructured streams of information
    - audio, video, telemetry
  - for information streams with “irrelevant” structures
Architecture

Basic building blocks

Combination rules

Recipes
Open Software Architecture

- to build effective useful software we must store information about:
  - how to perform the required functions
  - how the various parts relate to one another

- open systems must carry their own model so they can evolve, therefore
  - specifications of components must be visible from the system
  - implementations of components must be available to the system

. . . . . but how should this information be structured?
The (traditional) design process

1. Feasibility study
2. Specification
3. Design
4. Implementation
5. Test
6. Installation
7. Maintenance
Distributing the design process
Architecture is about structure

Raw materials → Composition rules → Conforming finished product
Component description

• by the process which created them . . .

• or . . . as components in their own right

  purpose: what is it for?

  meaning: how may it be used?

  structure: what other components does it need or consist of?

  guarantees: what promises will be kept?

  technology: what technologies are necessary?
Access to descriptions

Who?  
When?  
Why?
Summary

- everything should go faster, be more efficient, better tailored and less costly
- architecture is about structure: components and combination rules
- keep things separate and allow substitution
- keep information about components not their construction process
- make this information available as widely as possible

Are you ready to face the challenges?